Integrating with AMQP

Overview

Advanced Message Queuing Protocol (AMQP) is an open standard for passing business messages between applications or organizations. You can use Service Designer module to define an AMQP task. An AMQP task publishes the service request to an external system (Process Orchestrator) via a message broker.

The AMQP sends the service request to an exchange, which accepts the message from Service Catalog and routes them to message queues. RabbitMQ is an open source message broker software that implements the AMQP standard. An external system (Process Orchestrator) will access RabbitMQ with the required APIs to retrieve the messages and process them.

A new task type called Queue Service Request is available in the Service Designer module under the Plan tab to publish the service request data to a message broker. For more information on how to publish data using AMQP task, see Cisco Prime Service Catalog Designer Guide.

Message Queue

An exchange accepts messages from a producer application and routes them to message queues. The exchange that will be created for the pre, post, and the main AMQP tasks will be all of 'fanout' type with a default queue created for each of them and bound to each of them. The names of the default queues would be <topic-name>_queue. The topic name is the name that the user enters in the **Service Designer > Plan** tab for the 'Queue Service Request' task.

Note To prevent poodle attack, Prime Service Catalog integration with RabbitMQ server supports SSL protocol with TLSv1.2 version only. If TLSv 1.2 is not specified, then clients cannot connect to RabbitMQ server for consumption of messages and connections fail with this exception: javax.net.ssl.SSLException: Received fatal alert: protocol_version

Following is a sample code to create an exchange and queue and how to consume a message.

```
package amqpProject;


import com.rabbitmq.client.ConnectionFactory;
import com.rabbitmq.client.Connection;
import com.rabbitmq.client.Channel;
import com.rabbitmq.client.QueueingConsumer;
```

```
public class SampleProcess {

  public static void main(String[] argv) throws Exception {
    String exchangeName = "testExchange";
    String queueName = exchangeName+"_queue";
    String brokerIpAddress = "10.142.10.77";
    String userName = "admin";
    String password = "cisco123";
    factory.useSslProtocol("TLSv1.2"); !--Included to prevent Poodle attack
    ConnectionFactory factory = new ConnectionFactory();
    factory.setHost(brokerIpAddress);
    factory.setUsername(userName);
    factory.setPassword(password);

    Connection connection = factory.newConnection();

    Channel channel = connection.createChannel();
    channel.exchangeDeclare(exchangeName, "fanout", true, false, null);

    channel.queueDeclare(queueName, true, false, false, null);
    channel.queueBind(queueName, exchangeName, "");

    QueueingConsumer consumer = new QueueingConsumer(channel);
    channel.basicConsume(queueName, true, consumer);
    while (true) {

      QueueingConsumer.Delivery delivery = consumer.nextDelivery();
      String message = new String(delivery.getBody());
      System.out.println(" [x] Received from "+queueName+":-");
      System.out.println(message);

    }
  }
}
```

REST-based nsAPIs

There is a set of REST API for returning the message broker details for a service and its tasks. The signature and the response returned by it are given below. The input required is the service name that will be accepted as a path parameter. All the AMQP tasks for that service with the exchange details will be returned to the caller. In addition to the specific details about the service the generic information about the message broker such as the server IP, username, password to access the broker is also returned.

Given below is the sample response for a Service specific API:

`http://localhost:8088/RequestCenter/nsapi/messagebroker/service/<service name>`

```
{
   "rabbitmq_version":"3.2.4",
   "username":"guest",
   "password":"guest",
   "ipaddress":"10.42.9.56",
   "tasks":[
      {
         "name":"PRE:task1",
         "exchangeName":"exchange1",
         "payloadType":"Data, Form (medium-large)"
      },
      {
         "name":"task1",
         "exchangeName":"exchange2",
         "payloadType":"Data; No Service Details (default; small)"
      },
      {
         "name":"POST:task1",
         "exchangeName":"exchange3",
         "payloadType":"All Message Details (large)"
      }
   ]
}
```

The Overview API gathers all the exchanges that are configured in the Service Designer module and makes a call to find out what are the exchanges and queues that are already created in the RabbitMQ server. The Overview API performs a lookup on AMQP server to get information on queues and the output is sent across in JSON format.

Using Overview API and noCache parameter:

- When you specify RabbitMQ information through the UI and use the Overview API without noCache parameter, the information is retrieved from both cache and database.
- However, when you directly access database to insert RabbitMQ information, you must use Overview API with noCache =1 to ensure that the latest information is fetched.

The sample response from the RabbitMQ server is as follows:

```
http://localhost:8088/RequestCenter/nsapi/messagebroker/overview?noCache=1
{
"rabbitmq_version":"3.3.0","username":"admin","password":"iYqVYCDKbZTsuzc8wGKLvw==","ip
Address":"10.76.53.13","recoveryInterval":60000,"vhost":"/
","message-time-to-live":"300000","port":"5672","useSSL":true,"sslPort":"5671","ignoreC
ertificateError":true,"exchanges":
[{"name":"PO_Exchange_Pre","vhost":"/","type":"fanout","created":"false"},
{"name":"PO_Exchange_Primary","vhost":"/","type":"fanout","created":"false"},
{"name":"PO_Exchange_Post","vhost":"/","type":"fanout","created":"false"},
{"name":"newExchange1","vhost":"/","type":"fanout","created":"false"},
{"name":"newExchange2","vhost":"/","type":"fanout","created":"false"},
{"name":"newExchange3","vhost":"/","type":"fanout","created":"false"}],"queues":
```

[{"name":"PO_Exchange_Pre_queue","vhost":"/","created":"false"},
{"name":"PO_Exchange_Primary_queue","vhost":"/","created":"false"},
{"name":"PO_Exchange_Post_queue","vhost":"/","created":"false"},{"name":"newExchange1_queue","vhost":"/","created":"false"},
{"name":"newExchange2_queue","vhost":"/","created":"false"},{"name":"newExchange3_queue","vhost":"/","created":"false"}],
}

Encrypt Credentials using Public Key GUID

Prime Service Catalog supports a secure way to return AMQP credentials, using the GUID of the Public Key passed to the nsAPI.

If the external system Public Key GUID is passed as a Query parameter, then credentials are encrypted (using PO Secure String Format) in the response with the Public Key associated to the GUID

If the external system Public Key GUID is not passed, encrypted string in DB is returned as is (note: external system cannot decrypt this)

Encrypted sensitive data is returned in the following AMQP service specific and overview APIs:

http://localhost:8088/RequestCenter/nsapi/messagebroker/service/<service>?publicKeyGUID=..

http://localhost:8088/RequestCenter/nsapi/messagebroker/overview?publicKeyGUID=...

If the publickKeyGUID is not passed, or is incorrect, a validation error occurs and Http Code: 400 is returned.

Data Structure

The message format published is in the nsXml, similar to the format that is currently used by Service Link. The sample nsXml Data Structure is given below.

- Sample message structure for Data; No Service Details(default; small)

```
<?xml version="1.0" encoding="UTF-8"?>
<message xmlns:fo="http://www.w3.org/1999/XSL/Format" channel-id="">
<task-started task-type="task">
<task>
<actual-duration>0.0</actual-duration>
<completed-date/>
<context-id>470</context-id>
<context-type>Requisition Entry</context-type>
<due-date>2014-03-19 18:00:00</due-date>
<effort>10.0</effort>
<estimated-date/>
<expected-duration>10.0</expected-duration>
<flag-id>0</flag-id>
<is-sharable>true</is-sharable>
<is-shared>true</is-shared>
```

```
<next-action-id>2</next-action-id>
<performer-actual-duration>0.0</performer-actual-duration>
<priority>2</priority>
<scheduled-start-date>2014-03-18 16:00:00</scheduled-start-date>
<start-date>2014-03-18 10:22:25</start-date>
<state-id>2</state-id>
<subject>amqpTask1</subject>
<task-id>935</task-id>
</task>
<requisition>
<services>0</services>
<actual-cost>0.0</actual-cost>
<actual-duration>0.0</actual-duration>
<closed-on/>
<customer>
<company-address/>
<email>internal@newscale.com</email>
<fax/>
<first-name>admin</first-name>
<home-ou>
<name>&lt;s ID="847"/&gt;</name>
<organizational-unit-id>1</organizational-unit-id>
</home-ou>
<home-phone/>
<last-name>admin</last-name>
<login-name>admin</login-name>
<person-id>1</person-id>
<personal-address/>
<supervisor-name/>
<timezone>Pacific Standard Time</timezone>
<work-phone/>
</customer>
<due-on>2014-03-18 10:22:25</due-on>
<expected-cost>0.0</expected-cost>
<expected-duration>0.0</expected-duration>
<external>false</external>
<initiator>
<company-address/>
<email>internal@newscale.com</email>
<fax/>
<first-name>admin</first-name>
<home-ou>
<name>&lt;s ID="847"/&gt;</name>
<organizational-unit-id>1</organizational-unit-id>
</home-ou>
<home-phone/>
```

```
<last-name>admin</last-name>
<login-name>admin</login-name>
<person-id>1</person-id>
<personal-address/>
<supervisor-name/>
<timezone>Pacific Standard Time</timezone>
<work-phone/>
</initiator>
<organizational-unit>
<name>&lt;s ID="847"/&gt;</name>
<organizational-unit-id>1</organizational-unit-id>
</organizational-unit>
<requisition-entry>
<closed-date/>
<data-values>
<data-value multi-valued="false">
<name>amqpDict1.exchangeName</name>
<value>kk</value>
</data-value>
<data-value multi-valued="false">
<name>amqpDict1.queueName</name>
<value>pp</value>
</data-value>
<data-value multi-valued="false">
<name>amqpDict1.Message</name>
<value>ww</value>
</data-value>
</data-values>
<due-date>2014-03-19 18:00:00</due-date>
<item-number>1</item-number>
<price-per-unit>0.0</price-per-unit>
<priced>true</priced>
<quantity>1</quantity>
<rejected>false</rejected>
<rejected-date/>
<requisition-entry-id>470</requisition-entry-id>
<revision-number>5</revision-number>
<service>
<estimated-cost>0.0</estimated-cost>
<name>amqpService1</name>
<parameters>
<default-duration>0.0</default-duration>
<priority>2</priority>
<start-date/>
<start-mode>0</start-mode>
</parameters>
```

```
<pricing-schema>0</pricing-schema>
<quantity>1</quantity>
<service-id>29</service-id>
<version>5</version>
<standard-duration>0.0</standard-duration>
</service>
<start-after/>
<start-date>2014-03-18 10:22:25</start-date>
<start-mode>0</start-mode>
<status>1</status>
</requisition-entry>
<requisition-id>467</requisition-id>
<started-on>2014-03-18 10:22:25</started-on>
<status>1</status>
</requisition>
<context>
<requisitionentryref itemnumber="1"/>
</context>
</task-started>
</message>
```

- Sample message structure with secure strings in it.

```
<?xml version="1.0" encoding="UTF-8"?>
<message channel-id="F203ACC7-E6CA-A2EA-E040-007F0101140E"
xmlns:fo="http://www.w3.org/1999/XSL/Format">
<task-started task-type="task">
<task>
<actual-duration>0.0</actual-duration>
<completed-date/>
<context-id>69</context-id>
<context-type>Requisition Entry</context-type>
<due-date>2014-04-25 20:00:00</due-date>
<effort>10.0</effort>
<estimated-date/>
<expected-duration>10.0</expected-duration>
<flag-id>0</flag-id>
<is-sharable>true</is-sharable>
<is-shared>true</is-shared>
<next-action-id>2</next-action-id>
<performer-actual-duration>0.0</performer-actual-duration>
<priority>2</priority>
<scheduled-start-date>2014-04-24 18:00:00</scheduled-start-date>
<start-date>2014-04-23 14:11:13</start-date>
<state-id>2</state-id>
```

```
<subject>queueServiceRequest1</subject>
<task-id>266</task-id>
</task>
<requisition>
<services>0</services>
<actual-cost>0.0</actual-cost>
<actual-duration>0.0</actual-duration>
<closed-on/>
<customer>
<company-address/>
<email>internal@newscale.com</email>
<fax/>
<first-name>admin</first-name>
<home-ou>
<name>&lt;s ID=&quot;847&quot;/></name>
<organizational-unit-id>1</organizational-unit-id>
</home-ou>
<home-phone/>
<last-name>admin</last-name>
<login-name>admin</login-name>
<person-id>1</person-id>
<personal-address/>
<supervisor-name/>
<timezone>Pacific Standard Time</timezone>
<work-phone/>
</customer>
<due-on>2014-04-25 20:00:00</due-on>
<expected-cost>0.0</expected-cost>
<expected-duration>0.0</expected-duration>
<external>false</external>
<initiator>
<company-address/>
<email>internal@newscale.com</email>
<fax/>
<first-name>admin</first-name>
<home-ou>
<name>&lt;s ID=&quot;847&quot;/></name>
<organizational-unit-id>1</organizational-unit-id>
</home-ou>
<home-phone/>
<last-name>admin</last-name>
<login-name>admin</login-name>
<person-id>1</person-id>
<personal-address/>
<supervisor-name/>
<timezone>Pacific Standard Time</timezone>
```

```
<work-phone/>
</initiator>
<organizational-unit>
<name>&lt;s ID=&quot;847&quot;/></name>
<organizational-unit-id>1</organizational-unit-id>
</organizational-unit>
<requisition-entry>
<closed-date/>
<data-values>
<data-value multi-valued="false">
<name>amqpDict1.field1</name>
<value>a</value>
</data-value>
<data-value multi-valued="false">
<name>amqpDict1.field2</name>
<value>b</value>
</data-value>
<data-value is-secure="true" multi-valued="false">
<name>amqpDict1.field3</name>
<value>ut3u4RC699wz7Jd2O+4cix23m9XXgAC/EsDiHp1ERsOpdKSKdrtgMjFWcVoO96aCFSkCgwa2tkBo
vKoOHzaillNiJ47+aY8zCWKwd17tCjmMLEkeQlTvrDvIHR/DT1iWSmN08DI9+Ns7hY3A/g6ijUoM
gcuMczH+5F/pGtgupLZF/L7FwOwu4VcKVWM/2N5tuXGz+1aHTRAAAAByQXr/yD/75Ysy57LnQLxc
</value>
</data-value>
<data-value is-secure="true" multi-valued="false">
<name>amqpDict1.field4</name>
<value>ut3u4RC699wz7Jd2O+4cix23m9XXgABReynDp5AdBgRi5ivhS05Unv98BgWgxc5YNcZrCihhhH/1
rzZqhjiIjAoRelIjADDb3IQP72armXsLRTvh0/fusd4jLdVIm4q1s+GaSTt6F3oqQeZ4RLhVUopo
p2zNAxmjaGj629C8gWREes3Z8EjDvXg5K1YVi90fXJV1jDoADhAAAABP1hct5TNV2d8R1cN/qDtK
</value>
</data-value>
</data-values>
<due-date>2014-04-25 20:00:00</due-date>
<item-number>1</item-number>
<price-per-unit>0.0</price-per-unit>
<priced>true</priced>
<quantity>1</quantity>
<rejected>false</rejected>
<rejected-date/>
<requisition-entry-id>69</requisition-entry-id>
<revision-number>105</revision-number>
<service>
<estimated-cost>0.0</estimated-cost>
<name>queueService1</name>
<parameters>
<default-duration>0.0</default-duration>
```

```
<priority>2</priority>
<start-date/>
<start-mode>0</start-mode>
</parameters>
<pricing-schema>0</pricing-schema>
<quantity>1</quantity>
<service-id>2</service-id>
<version>105</version>
<standard-duration>0.0</standard-duration>
</service>
<start-after/>
<start-date>2014-04-23 14:10:48</start-date>
<start-mode>0</start-mode>
<status>1</status>
</requisition-entry>
<requisition-id>64</requisition-id>
<started-on>2014-04-23 14:10:47</started-on>
<status>1</status>
</requisition>
<context>
<requisitionentryref itemnumber="1"/>
</context>
</task-started>
</message>
```